HARDWARE PROTOTYPING OF TWO-WAY RELAY SYSTEMS

A Thesis

by

QIONG WU

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2012

Major Subject: Electrical Engineering

HARDWARE PROTOTYPING OF TWO-WAY RELAY SYSTEMS

A Thesis

by

QIONG WU

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Shuguang Cui
Committee Members,	Jean-Francois Chamberland-Tremblay
	Srinivas Shakkottai
	Riccardo Bettati
Head of Department,	Costas Georghiades

August 2012

Major Subject: Electrical Engineering

ABSTRACT

Hardware Prototyping of Two-Way Relay Systems. (August 2012)

Qiong Wu,

B.S., Beijing University of Posts and Telecommunications

Chair of Advisory Committee: Dr. Shuguang Cui

In this thesis, I conduct the hardware prototyping of a two-way relay system using the National Instruments FlexRIO hardware platform. First of all, I develop several practical mechanisms to solve the critical synchronization issues of the systems, including Orthogonal Frequency-Division Multiplexing (OFDM) frame synchronization at the receiver, source to source node synchronization, and handshaking between the sources and relay nodes. Those synchronization methods control the behaviour of the two source nodes and the relay node, which play critical roles in the two-way relay systems. Secondly, I develop a pilot-based channel estimation scheme and validate it by showing the successful self-interference cancellation for the two-way relay systems. In particular, I experiment the self-interference cancellation technique by using several channel estimation schemes to estimate both source to relay channels and relay to source channels.

Moreover, I implement the physical layer of a 5 MHz OFDM scheme for the two-way relay system. Both the transmitter and receiver are designed to mimic the Long Term Evolution (LTE) downlink scenario. The physical layer of the transmitter has been implemented in Field-Programmable Gate Arrays (FPGAs) and executed on the hardware board, which provides high throughput and fundamental building blocks for the two-way relay system. The physical layer of receiver is implemented in the real-time controller, which provides the flexibility to rapidly reconfigure the system. Finally, I demonstrate that the 5MHz OFDM based two-way relay system can achieve

reliable communications, when the channel estimation and system synchronization can be correctly executed.

To my parents

ACKNOWLEDGMENTS

The time I spent in Aggieland is one of the most exiciting and rewarding periods in my life. I owe my gratitude to all those people who have made this thesis possible and also made my life in Aggieland the one that I will cherish forever.

First of all, my deepest gratitude goes to my advisor, Dr. Shuguang Cui, for his continuous support and guidance throughout these years. I have been fortunate to learn from him, not only the immense knowledge but also how to question and express ideas. More importantly, his motivation, enthusiasm, and professional attitude have always been my learning examples.

I am also grateful to the rest of my committee members, Dr. Jean-Francois Chamberland-Tremblay, Dr. Srinivas Shakkottai, and Dr. Riccardo Bettati. I really enjoy the class that I took from Dr. Jean-Francois Chamberland-Tremblay and Dr. Riccardo Bettati, who are willing to stop and explain every details in the classroom. I am also thankful to Dr. Srinivas Shakkottai for being my committee member and responding all my requests promptly.

I am deeply indebted to all current and past group members, including Meng Zeng, Jiaming Qiu, Qing Zhou, Chuan Huang, Amin Banaei, Amir Salimi, Jianwei Zhou, Di Li, Hang Li, Liang Ge, and also our visiting scholars including Professor Rui Ma and Yang Zhou. I also want to especially thank Lili Zhang, who helps me and teaches me a lot in these two years.

Lastly, but most importantly, none of this would be possible without the love and support of my family. I wish to express my deepest love to my parents, who raised me and gave me abundant freedom and trust to make every important decisions by myself.

TABLE OF CONTENTS

CHAPTER Page

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

One target of communication system design is transmitting and receiving information at a higher throughput with a lower bandwidth. The network coding concept is one of the promising techniques to achieve that [1]. In particular, by allowing intermediate network nodes to mix the data or signals received from multiple links, as opposed to separating them by traditional approaches, network coding reduces the amount of transmissions in the network and thus improves the overall network throughput.

Recently there have been many research topics focused on the area of network coding. One of them is two-way relay network systems. In the modern network, more and more interests have been placed on utilizing relays. The two-way relay network system is a good example, which utilizes the idea of network coding and self-interference cancellation. This thesis follows the theoretical results in [2]. Specifically, it is assumed that the two source nodes are equipped with a single antenna each and the relay is equipped with multiple antennas. The optimal beamforming matrix at the relay node is studied and the optimal relay beamforming structure as well as an efficient algorithm to compute the optimal beamforming matrix based on convex optimization techniques is given. Since there is a gap between the theoretical and practical performance, one target of this thesis is trying to quantify the gap by prototyping this two-way relay system with realistic setup. In addition, we plan to evaluate its potential in the industrial applications. Another target of this thesis is trying to identify new problems that are important in practical implementations.

The two-way relay system is one of the basic elements in decentralized/centralized

---

This thesis uses the journal model of *IEEE Transactions on Automatic Control*.

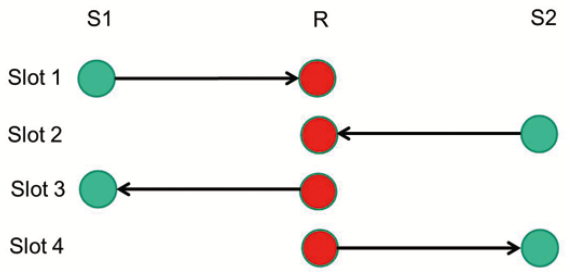

Fig. 1.: Four Time Slots Communication Scheme

wireless networks. The simplest two-way relay system that we prototype consists of two source nodes and one relay node. The two source nodes exchange messages through the aid of the relay node. Traditionally, in order to achieve the message exchange and avoid interference between two source nodes, one orthogonal communication scheme could be used, which is Time-Division Duplexing (TDD) and takes four time slots to finish the two=way transmission, as shown in Fig. 1. In the first two time slots, the two source nodes transmit messages to the relay. In the last two time slots, the relay node sends the received messages back to the two source nodes, respectively.

However, by applying the concept of network coding and self-interference cancellation, the total number of time slots could be reduced from four to two without using extra frequency resources. As shown in Fig. 2, the two source nodes transmit information to the relay at exactly the same time over same frequency spectrum. In the first time slot, the relay receives the superimposition of the signals from the two source nodes. In the second time slot, the relay directly amplifies and forwards this superimposed signal back to both source nodes. After receiving the signal from the relay, since the two source nodes know exactly what they have transmitted to the relay in the first time slot, they could cancel the self-interference by subtracting their own message out of the received signal. Therefore, each source node could easily decode the information from the other source node. Note that, in this communication

scheme, we apply the network coding concept, with the self-interference cancellation method employed.

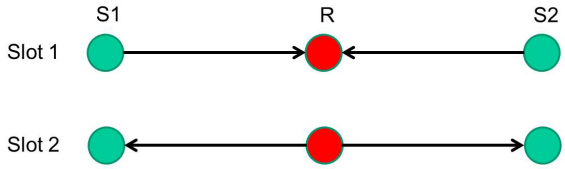

Fig. 2.: Two Time Slots Two-Way Relay Communication Scheme

In this thesis, we prototype the two-way relay system by using the National Instrument programmable wireless nodes. The software platform used for the implementation consists of LabVIEW, LabVIEW FPGA, and LabVIEW Real-Time developed by National Instruments. LabVIEW is a graphical programming language, where all the logics and subroutines are mapped as blocks, and their input and output are interconnected by virtual cables in the graphic diagrams. It is also a data flow and parallel programming language. The program execution follows the data flow through the virtual cables and blocks. Moreover, when a branch is reached in the virtual cables, these branches are executed concurrently. The hardware platform we adopt is the National Instrument Lead User Setup, which contains NI PXIe 1071 hardware chassis, NI PXIe 8130 real-time controller, and the FlexRIO platform integrating with a NI 5781 base-band transceiver combined with an NI PXIe-7965R FPGA module. The hardware and software platform aim to enable rapid prototyping of communication and signal processing systems. The algorithms and logics can be easily expressed in a purely graphical code and directly mapped to hardware operations or lower level hardware description languages, where the LabVIEW FPGA is capable of converting its LabVIEW graphical programming language to VHDL codes, and it also provides a large library of APIs that can be directly dragged and compiled into hardware bitfiles.

CHAPTER II

SYSTEMS MODEL

The two-way relay system aims to exchange the information through the aid of the relay with high efficiency. It utilizes the concept of analog network coding and self-interference cancellation. The channel model is shown in Fig. 3. The source nodes S1 and S2 attempt to exchange information via the relay node R. In the first time slot, S1 and S2 transmit their messages to the relay simultaneously. Therefore, the relay receives the mixture of these two incoming signals. In the second time slot, the relay directly amplifies and broadcasts the superimposed signal to both S1 and S2. Since the S1 and S2 know exactly what they transmitted to the relay R in the first time slot, the two source nodes subtract their own message first and then decode the information from the other side, which is called self-interference cancelation. In our prototyping systems, both of the two source nodes are equipped with a single antenna and the relay node is also equipped with a signal antenna. Therefore, all of the transmitted symbols are expressed as scalars. We denote the message transmitted at the S1 as X1 and the message transmitted at S2 as X2. The two source to relay channels in the first time slot are denoted as h_1 and h_2, respectively. And the two relay to source channels are denoted as h_3 and h_4, respectively. The received baseband signal at the relay R in the first time slot is expressed as

$$Y_R = h_1 x_1 + h_2 x_2 + z_1. \tag{2.1}$$

The first term on the right hand side of (2.1) is the signal transmitted from the S1, which experiences the source to relay channel h_1. The second term on the right hand side of (2.1) is the signal transmitted from S2, which experiences the source to relay channel h_2. The last term z_1 is the received noise at relay, and it is assumed

that $z_1(n) \sim CN(0,1)$. Upon receiving this superimposed signal, the relay processes the received signal with amplify and forward relay operation, and then broadcasts this amplified signal to S1 and S2 during the second time slot. In this model, we do not assume channel reciprocity for the two-way relay channel in the source to relay links and relay to source links. Thus, the received signals at the S1 is expressed as:

$$Y_1 = h_3 h_1 x_1 + h_3 h_2 x_2 + h_3 z_1 + z_2. \tag{2.2}$$

where z_2 is the noise at S1, and it is assumed that $z_2(n) \sim CN(0,1)$. On the right hand side of (2.2), the first term is the self-interference from S1 and the second term contains the interested messages from S2. Assuming that our platform could estimate the channel paramaters h_1, h_2, h_3, S1 could firstly subtract the self-interference term $h_3 h_1 x_1$, and then decode the second term $h_3 h_2 x_2$. Therefore, the message from S2 can be correctly decoded at S1. Similar operations are executed at S2.

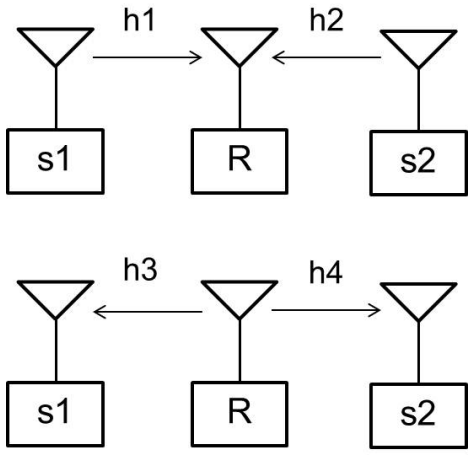

Fig. 3.: Two-Way Relay System Model

CHAPTER III

SYSTEMS DESIGN

In this chapter, we provide an overview of the system design, including the software and hardware descriptions and hardware configurations, introduction to the master-slave relationship, targeted systems and several implementation challenges.

A. Software Descriptions

The software platform that we use to prototype this system consists of LabVIEW, LabVIEW FPGA, and LabVIEW Real-Time. LabVIEW is a graphical system design software which enables rapid prototyping of communications and signal processing algorithms. Through LabVIEW, a designer can connect to the hardware setup and can perform algorithms design and simulations as well as real-time implementation using the same software environment and tools. LabVIEW FPGA and LabVIEW Real-Time are two subsets of LabVIEW, which are particularly useful for the implementation of algorithms for FPGA and processing time critical applications. The applications written in the LabVIEW is called virtual instruments. As shown in Fig. 4, every application has two types of representation diagrams. One is called the front panel, which is the virtual front panel of the virtual instruments, and mainly contains the input and output of the application. Another representation is the block diagram, which is the internal logic of the virtual instruments, including operations, algorithms, and subroutines. Instead of writing text-based code, the programmer drags the logic blocks, symbols, and subroutines onto the block diagram, where those blocks are connected through virtual cables to represent input and output flows. One inherent nature of programming in LabVIEW is that the branches in the block diagram are running in parallel. This feature is very useful in programming for FPGAs.

The programmer can choose whether to run the whole algorithm or part of it on a host PC, real-time operating system (OS) with dedicated processors, or a FPGA, based on the computational complexity and latency requirements of the algorithm.

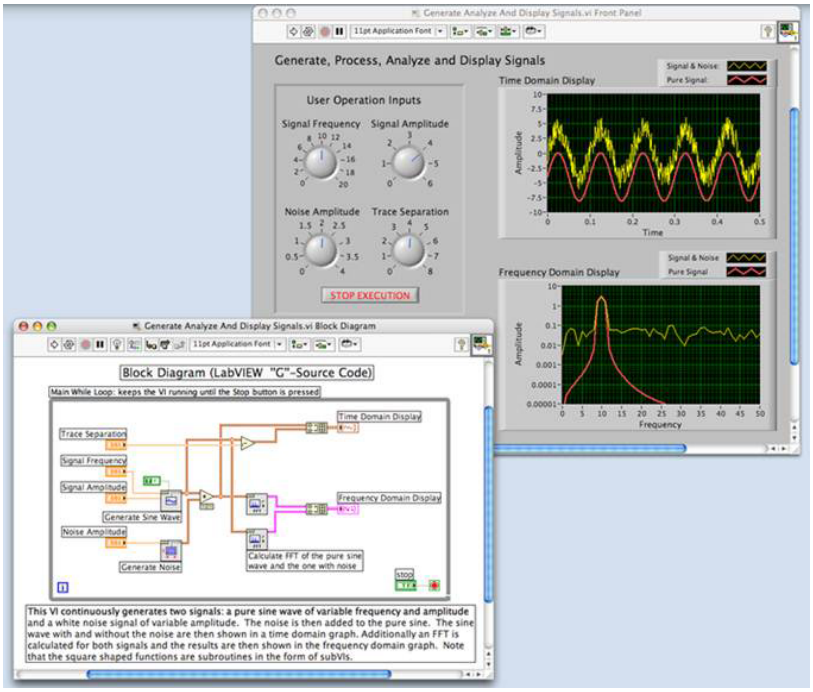

Fig. 4.: LabVIEW

The LabVIEW FPGA enables the ability of programming in FPGA in the LabVIEW graphical programming fashion. All the LabVIEW FPGA code is LabVIEW-based graphical code. The LabVIEW FPGA code mimics the operations of the FPGA circuit. The logic and subroutine blocks in the FPGA stay inside a 'signal cycle time loop'. It is a loop running on a specific clock rate according to the application requirements. The blocks inside the loop are set to run at this specific clock rate. The LabVIEW FPGA also has the parallel programming feature. The blocks in the parallel branches in the single cycle time loop will be running in parallel. The LabVIEW FPGA code can also run on a host PC in the simulation mode with LabVIEW platform for testing. It mimics the operations of the FPGA but not runs exactly the same

as in the FPGA. For example, the running clock rate of the simulation mode is not the same as running in FPGA. Moreover, the logic delay and routing delay cannot be tested in the simulation mode. Once the code passes the simulation mode test, it could be directly compiled. During the compilation, several issues will be checked in the process, including the timing violation of the circuit, the resource utilization of the FPGA, and the FPGA input output interface. The compiled LabVIEW FPGA code produces a bitfile, which can be directly loaded into the FPGA and executed on the platform.

The LabVIEW Real-Time guarantees to finish certain operations or processing within a specific time interval. This feature is achieved by allocating a dedicated processer to a specific task, where a real-time operating system is used to measure whether the timing requirements are satisfied or not.

B. Hardware Descriptions

We prototype the two-way relay system by using National Instruments Lead User Setup, which consists of several important components as follows. It includes NI PXIe 1071 hardware chassis, the NI PXIe 8130 real-time controller, and the FlexRIO which has NI 5781 base-band transceiver combined with an NI PXIe-7965R FPGA module, and an RF front end as Ettus XCVR2450. The details about the equipments are given in the following subsections, based on the information from [5] :

1. NI PXIe 1071

The hardware chassis is shown in Fig. 5. The NI PXIe 1071 chassis kit consists of a low-cost, compact 4-slot chassis featuring a controller slot, which accepts either an embedded controller or a remote controller, and three peripheral slots. The NI PXIe

1071 offers the flexibility to populate each peripheral slot with either a PXI Express module or PXI module. The hardware chassis is used to contain all the hardware and provides high speed communication interface between those equipments. It also has 10MHz and 100Mhz on board reference clocks with low jitter, which is used in our synchronization scheme.

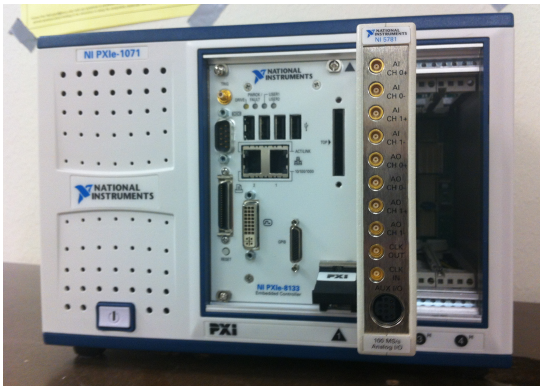

Fig. 5.: NI PXIe 1071

2. NI PXIe 8130

The NI PXIe 8130 is a high-performance AMD Turion 64 X2 processor-based embedded controller in PXI Express systems, with a 2.3 GHz dual-core processor and dual-channel 667 MHz DDR2 memory. It also has up to 4 Gb per second bandwidth interface to the chassis kit. As shown in Fig. 6, it also equipped with several Ethernet ports that allows various protocols to run inside the system. Our application takes advantages of these features to solve the handshaking problem between the source nodes and relay node.

3. NI FlexRIO 5781 and NI FlexRIO 7965R

As shown in Fig. 7, the NI 5781 is an analog dual-input, dual-output NI FlexRIO adapter module optimized for interfacing with baseband-to-RF upconverters and

Fig. 6.: NI PXIe 8130

downconverters. When it is paired with an NI FlexRIO FPGA module, the resulting NI 5781 is an FPGA-enabled RIO baseband transceiver that can be used to implement custom RF modulation and demodulation, channel estimation, bit error rate testing, and spectral monitoring and jamming. In our systems, the NI 5781 is mainly used as a baseband signal acquisition and transmission interface, which contains Analog to Digital Converters (ADCs) and Digital to Analog Converters (DACs) with sampling rates up to 100 M samples per second at 14 and 16 bits resolution level respectively. The NI FlexRIO 7965R is the PXI Express field-programmable gate array (FPGA) module, which can be coupled with the I/O adapter modules. Programmed with the NI LabVIEW FPGA module, these modules together provide high-performance I/O and user-defined hardware processing on the PXI platform. The NI 7965 is the core signal processing unit in the prototyping systems, which contains a Xilinx Vertex 5 FPGA and 512 MB on-board DRAM.

Fig. 7.: NI FlexRIO 5781 and NI FlexRIO 7965R

4. Ettus XCVR2450

The XCVR2450 is a high-performance transceiver intended for operation at 2.4 GHz and 5.9 GHz range, which is shown in Fig. 8. Filtering on the XCVR2450 provides good selectivity and large dynamic range in the intended operation bands. In our systems, the XCVR2450 operates at 2.4GHz, which is in the ISM band. The typical power output and noise figure of the XCVR2450 are 100 mW and 8 dB, respectively. Furthermore, this RF font-end can switch rapidly between transmission and reception mode, which provides support for the TDD operation in the two-way relay system.

Fig. 8.: Ettus XCVR2450 RF Front End

C. Hardware Configurations

In the two-way relay system, there are three nodes: two source nodes and one relay node. In our project, the three nodes are configured as Fig. 10: S1 and S2 are located in the same chassis (source chassis), and the relay node is located in another chassis (relay chassis). Specifically, we put them in the same chassis to share a common clock, in order to solve this synchronization problem, such that both source node 1 and 2 could be triggered and synchronized within several nanoseconds. In Fig. 9, we show a real scenario hardware configuration in our lab. For convenience, we connect the source chassis with the relay chassis by Ethernet to transmit some control messages for synchronization: e.g., TX/RX trigger messages, which control the TX/RX behavior of the sources and relay nodes.

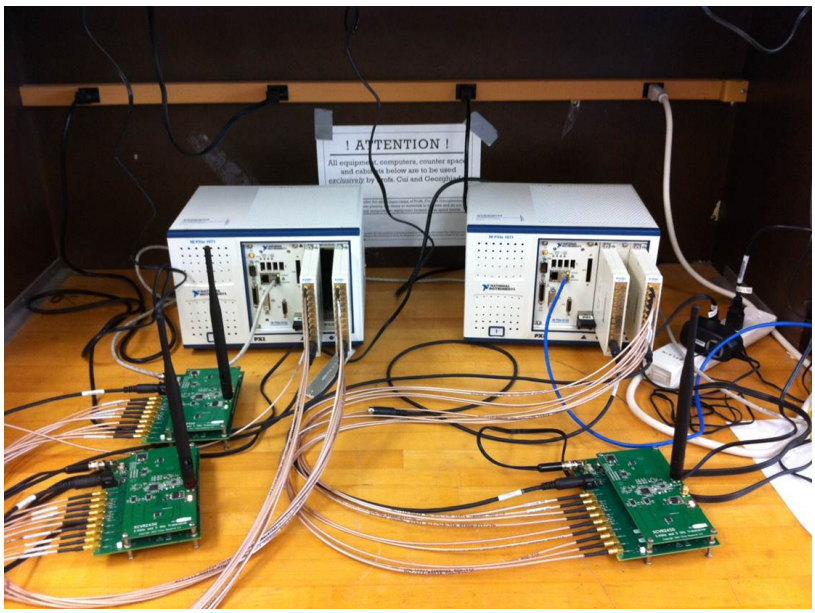

Fig. 9.: Hardware Setup in Real Environment

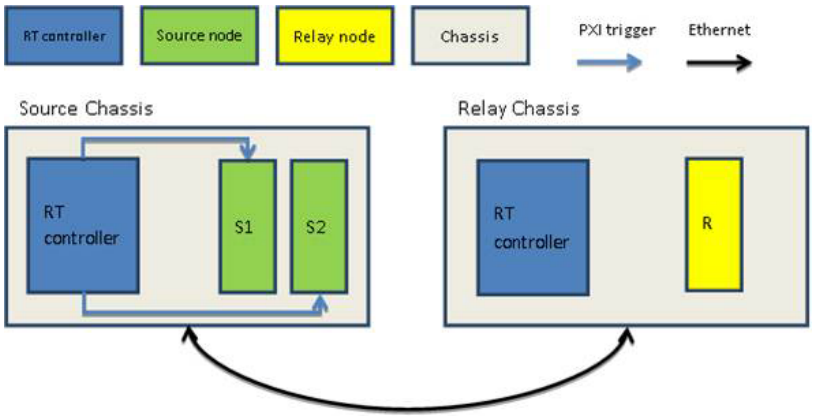

Fig. 10.: Hardware Configuration

D. Master/Slave Relationship

In this project, we follow a master/slave design paradigm to control the behavior of the source chassis and relay chassis. With this model, we can achieve the handshaking and TDD time slot synchronization as follows: The source chassis is the master node and the relay chassis is the slave node. In each time slot, the master node switches between TX and RX modes according to the instructions from the RT controller and then it sends trigger messages through Ethernet to trigger the slave node to switch between RX and TX modes, respectively. For example, at the beginning of the TX time slot for the master node, it sends a trigger message to the slave node to trigger its RX time slot. After the master is confirmed by the slave, it starts the transmission. The same procedure follows in the RX time slot for the master node.

E. Targeted Systems

In our targeted prototype two-way relay system, both of the two source nodes are equipped with a signal antenna and the relay node is also equipped with a signal antenna. The physical layer of the systems is designed to mimic the 5 MHz bandwidth LTE downlink scenario. The number of subcarriers is 300 and the subcarrier spacing

equals to 15 KHz. The modulation scheme used in this prototype is the Quadrature Amplitude Modulation (QAM). The data throughput is 6Mb per second and system sampling rate equals to 7.68MHz. For the two-way relay system, the length of the time slot is configurable ranging from several milliseconds to several seconds.

F. Challenges

Although conceptually two-way relaying sounds simple, there are several challenges to the implementation of the system. The first one is that the symbol level synchronization needs to be achieved in such that the source to relay signals are correctly superimposed with each other. The second one is that all the channel state information should be precisely estimated at all the three nodes. The third one is that the three chassis need to be synchronized with each other in the first and second time slots, where the beginning of each TX and RX time slot should be perfectly aligned. In the following subsections, we give more details about the above challenges.

1. Channel Estimation

In order to achieve successful message exchanging between the two source nodes, the system is required to estimate the channels h_1, h_2, h_3 and h_4 as accurate as possible. Specifically, S1 needs to know channel state information h_1, h_2 and h_3 and S2 needs to know channel state information h_1, h_2 and h_4. In the traditional channel estimation scheme for the LTE downlink scenario, the pilot is deployed to estimate the channel. In the frequency domain, the pilot is interleaved with the data symbols: Every 5 data symbols are packed together with 1 pilot symbol. It is assumed that the channel stays flat over every 6 symbols in frequency domain. After receiving these symbols, the channel is estimated by dividing the received signal level with the transmitted

signal level. However, this pilot estimation scheme is not suitable in our two-way relay system due to the following reasons. In the first time slot, the two incoming signals at the relay are superimposed with each other, where the pilots are polluted due to the superimposing operations. Meanwhile, in order to estimate the channel in the first time slot, we need to estimate h_1 and h_2 at the same time.

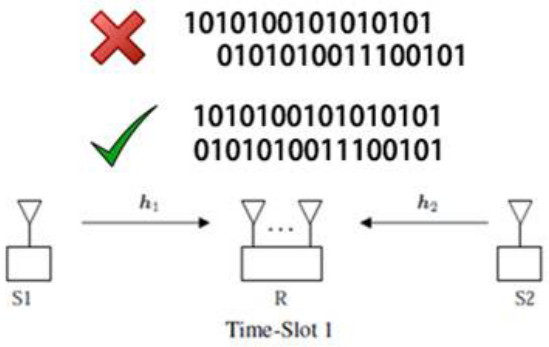

Fig. 11.: Source to Source Synchronization

2. Source to Source Synchronization

Another challenge is to achieve the synchronization between the two source nodes. In the two-way relay system, the two source nodes need to transmit the message symbols at exactly the same time, such that they are correctly superimposed with each other when the messages arrive at the relay. As shown in Fig. 11, if there is a small offset between these two symbols, the two messages will be severely inferencing with each other. Therefore, the source to source synchronization is an important requirement in the two-way relay system. Specifically, in the first time slot, the two source nodes transmit their messages to the relay, where the two incoming messages have to be correctly superimposed with each other. Therefore, the messages need to be synchronized symbol by symbol over the air. Then in the second time slot, the relay broadcasts the superimposed messages back to source nodes, where the self-

inference cancellation can be correctly performed. Simply speaking, there are two ways to achieve this synchronization. The first one is trying to align the beginning of the two source to relay messages in the first time slot and then in the second time slot, we directly cancel the self-interference at the beginning of each received message at the source nodes. The second method is to estimate the offset of the two source-to-relay messages, and in the second time slot, we start to cancel the self-interference from the offset index. In our prototyping systems, we employ the first mechanism for a better performance.

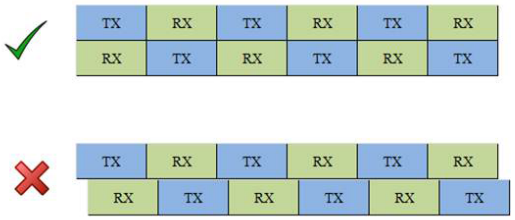

Fig. 12.: Source to Relay Synchronization

3. Source to Relay Synchronization

The source to relay synchronization is also one of the challenges during implementation. Specifically, the beginnings of the TX time slot at the sources and the RX time slot at the relay should be perfectly aligned such that the systems do not suffer from the sample loss. If it is not perfectly aligned, there exist a time period where both of the antennas are receiving or transmitting which results in message loss, as shown in Fig. 12. Specifically, in the first time slot, the two source nodes start to transmit messages to the relay and the relay needs to start its reception at the same time. If the relay starts the reception later than the time when the two source nodes start to transmit, the relay node misses a portion of the messages transmitted from the two source nodes. In the second time slot, the relay node starts to broadcast the messages

to the source nodes. The messages transmitted by the relay can also be lost, if the relay starts to transmit too early. Therefore, the handshaking between the source nodes and the relay node is important for the implementation of the two-way relay system. The accuracy requirement of the source to relay synchronization is related to the time slot duration. In our system, we are targeting at a configurable time slot duration ranging from multiple milliseconds to several seconds. Accordingly, the synchronization requirement of the source to relay synchronization should be set at the sub-millisecond level.

4. Self-Interference Cancellation

Self-interference cancellation at the source nodes is another challenge in the two-way relay system. As shown in the previous chapter, in order to perform self-interference cancellation, the sources need to know the channel state information as accurate as possible. Specifically, S1 needs to know the channel gain h_1, h_2, and h_3. And S2 needs to know the channel gain h_1, h_2, and h_4. An error in the channel estimation will lead to decoding errors in the messages.

CHAPTER IV

IMPLEMENTATION DETAILS

In this chapter, we provide a description to the implementation details and also propose our solutions to the challenges discussed in the previous chapter.

A. Physical Layer

The physical layer of the two-way relay system follows the LTE downlink design. The block diagram for the transmitter is shown in Fig.13. At the beginning, the messages are generated from a random message generator in the RT controller with transmission throughput 6Mb/second. Then the messages are modulated into QAM symbols and the throughput is reduced to 3M symbols/second. Every frame of messages is over 250 data subcarriers. After the modulation, the message symbols are interleaved with the pilots, where the throughput is raised from 3M to 3.6 M symbols/second, where every 5 data symbols are interleaved with 1 pilot. Thus, each frame contains 250 data subcarriers and 50 pilots subcarriers. Afterwards, the interleaved message are passed to the zero padding block to pad zeros around the frame. The objective of the zero padding is to align the frame size to a proper IFFT size. In our case, the zeros are padded around each frame to execute a 512-point IFFT operations. Then these data go through the IFFT block with cyclic prefix (CP) insertion to generate the LTE symbol format. We follow the LTE long CP configuration, where the CP length equals 128. The IFFT and CP insertion operations further increase the frame length from 512 to 640. The throughput after CP insertion is raised to 7.68M symbols/second. Finally, these symbols are resampled to 10M symbols/second to meet the clock rate requirement and then passed to the RF front-end.

For the receiver, it mainly operates in the reverse direction of the transmitter,

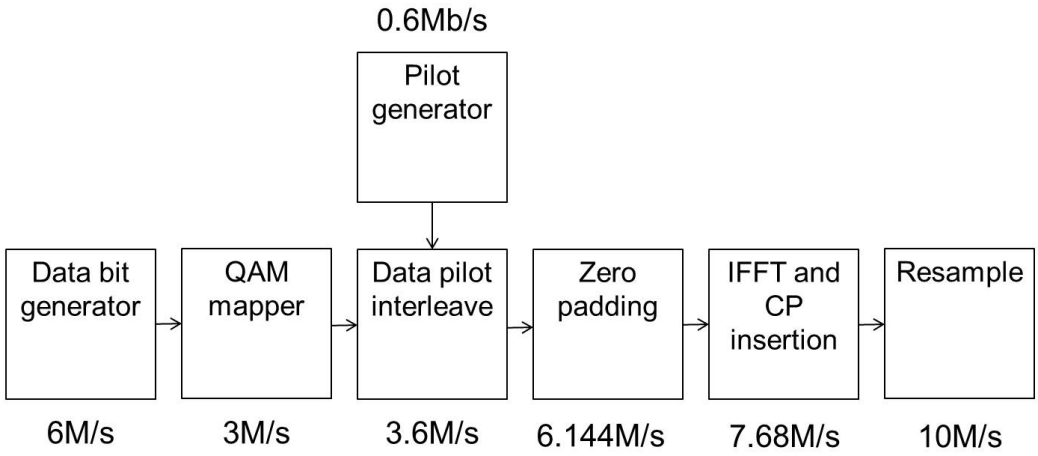

Fig. 13.: Transmitter Physical Layer Block Diagram

which is shown in Fig. 14. The main difference between the receiver and the transmitter is that the receiver contains an OFDM frame detection block, which is deployed to detect the beginning index of the OFDM frame. Another difference is that the receiver also includes a channel estimation block after extracting the pilots out from the received OFDM symbols.

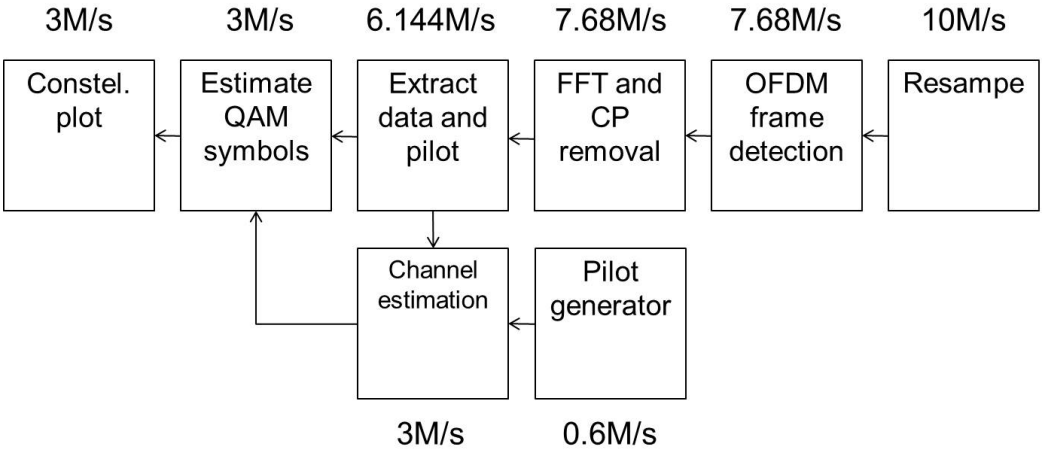

Fig. 14.: Receiver Physical Layer Block Diagram

How to map these blocks into the hardware is shown in Fig. 15. For the transmitter side in our current setup, except for the data and pilot generation blocks, all

other blocks stay in the FPGA. The data and pilot generation blocks are located in the RT controller. For the receiver, the current demo system is operating on a non-realtime fashion. Only the resampler is implemented in the FPGA and all other blocks are implemented in the RT controller.

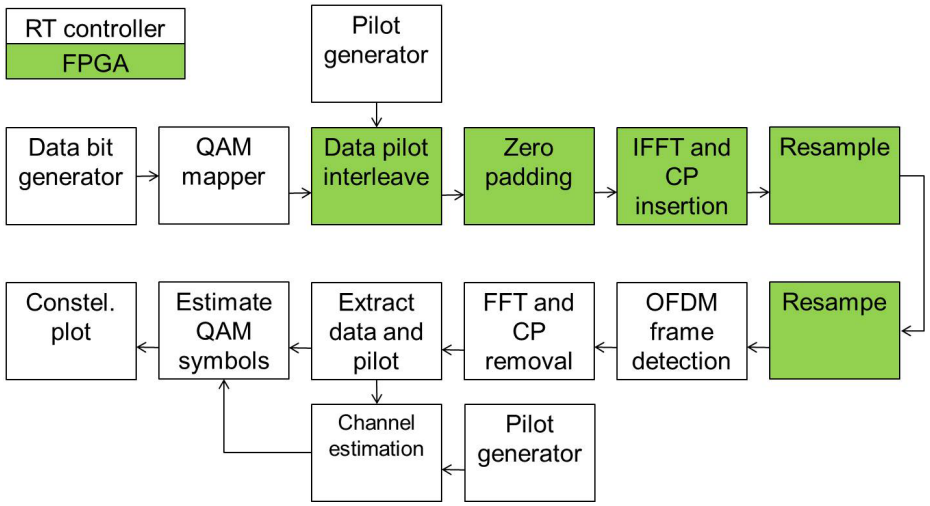

Fig. 15.: Hardware Mapping of Physical Layers

B. Master/Slave Structure

The master and slave need to be synchronized such that the source and relay chassis can correctly communicate with each other in the TDD system. The synchronization issue is solved in the following way: When the master is in TX time slot, it sends a trigger RX message to the slave. Similarly, when the master is in the RX time slot, a trigger TX message is transmitted to the slave. The Master/Slave synchronization uses the UDP messages transmitted between the master and slave nodes, and the slave can be triggered to switch the antenna mode according to the master node behavior. For both the master and slave nodes, the TX/RX switching is controlled by the RT controller. As shown in Fig.16, the RT controller in the source chassis acts as the the master and all the other equipments are slaves. The slave nodes receive trigger

messages from the master node through various ways according to different requirements. The two source nodes in the source chassis are controlled through a hardware trigger in the FPGA, similarly for the relay in the relay chassis. For the source to relay synchronization, we use the software triggers, since the accuracy requirement is lower compared with the source to source synchronization. The trigger message is processed and generated by the RT controller within millisecond and transmitted through the Ethernet.

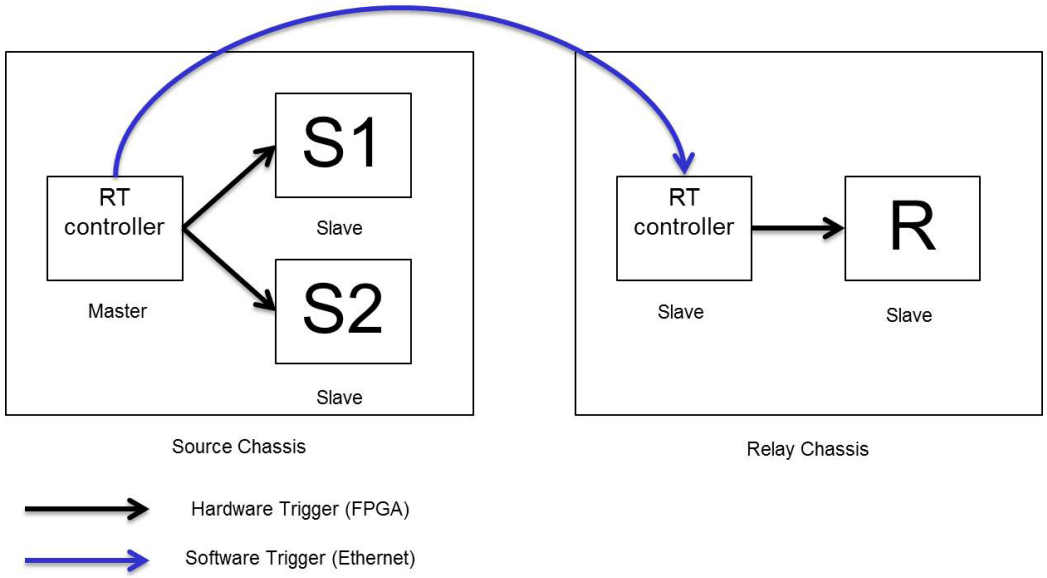

Fig. 16.: Master/Slave Structure

C. Antenna TX/RX Mode Switching

The antenna switching needs to run fast enough such that the latency related to the switching does not affect the two-way relay system performance. In XCVR2450, it uses a MAXIM 2829 chip as the controller. The switching speed of the chip between TX and RX modes is around 1 microsecond. Moreover, the antenna switching block is implemented as a state machine in the FPGA. By writing instruction bits into the

2829 chip, the antennas in both the relay and source chassis can switch between TX and RX modes. When an external trigger message is received, the FPGA can switch the TX/RX mode of the antenna according to instruction, and the TX/RX switching can be finished within microsecond level.

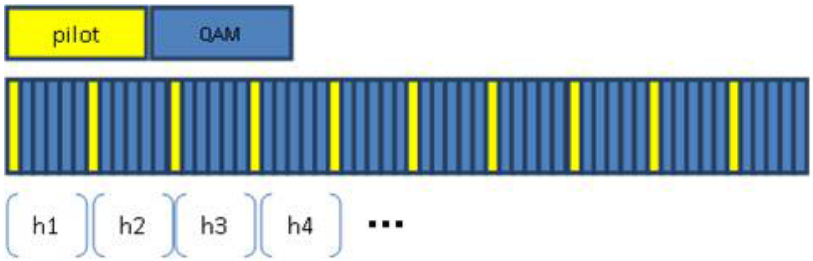

Fig. 17.: Traditional Channel Estimation Scheme in LTE Downlink Scenario

D. Channel Estimation Scheme

In LTE downlink scenario [4], the pilot is used to estimate the channel, with each pilot symbol followed by 5 data symbols as shown in Fig. 17, where we assume that the channel remains flat over every 6 symbols. In out setup, the frame structure follows the LTE standard with 5 MHz bandwidth and a long CP configuration. After receiving those symbols, we calculate the channel parameters [h_1, h_2, h_3, h_4] in Fig. 17 by dividing the received pilot symbol level with the transmitted symbol level. Afterwards, we can use the estimated channel parameters to decode the remaining 5 received data symbols.

However, the traditional channel estimation scheme in the LTE downlink scenario does not work for our two-way relay system, since it only estimates the channel from a single user. In order to estimate the two source-to-relay channel at the same time, we propose a new channel estimation scheme by modifying the frame structure, which is shown in Fig. 18. Specifically, we use two pilots in every 6 symbols. One is

coming from S1 and another is from S2. If we assume these two messages are synchronized in the transmission, we could assign an empty symbol when the other node is transmitting pilot in that particular subcarrier. In other words, we assign an empty symbol at the S1 when S2 is transmitting the pilot and vice versa. Here we share the frequency spectrum of the OFDM systems and try to allocate the pilots in different subcarriers without mixing them, since the pilots from two sources are orthogonal with each other. Note that, in order to obtain the channel state information from both source nodes, we trade the bandwidth for the channel state information. For the traditional channel estimation scheme in the LTE downlink scenario, every 5 data subcarriers are interleaved with 1 pilot, only 16% of the bandwidth is used to estimate the channel. In our channel estimation scheme, in order to estimate the source to relay channels in the first time slot at the same time, we allocate another data subcarrier to the pilot symbol. Therefore, about 33% of the bandwidth is utilized to estimate the channel h_1 and h_2 in the first time slot.

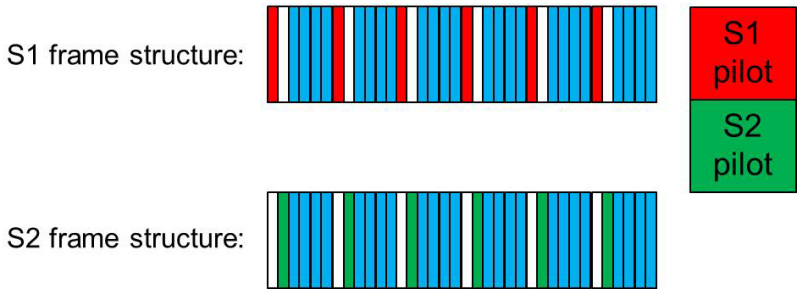

Fig. 18.: Channel Estimation Scheme in Two-Way Relay Systems

E. The Channel State Information at the Sources

Another issue for the channel estimation scheme is how we could make the two source nodes know the channel state information at the relay. One obvious solution is to estimate the channel at the relay and then feed it back to the sources. However, there

are also several issues with this solution. First, if the feedback method is used, we need to decide which channel to use for feeding channel state information back to the source nodes. One way is to use the Ethernet that we use to coordinate the source to relay synchronization. We can take advantage of this feature again and feed the channel state back through the Ethernet. However, one issue related to this method is how the channel state and message could be correctly synchronized with each other symbol by symbol, which is hard to achieve since there is a non-deterministic delay for the message transmission over Ethernet, where the synchronization requirement for the channel state and message is around 20 nanoseconds.

In order to avoid the above issue, we propose another mechanism to solve this problem. Instead of feeding the channel state from the source to the relay through Ethernet, here we insert the unestimated pilots of h_1 and h_2 into the relay-to-source message as shown in Fig. 19. Therefore, the source could extract and process these unestimated pilots locally and render the channel information at the relay. Specifically, in the proposed channel estimation scheme, 100 subcarriers are assigned to the pilot subcarriers and there are in total 300 subcarriers in a single frame. When the relay broadcasts the superimposed messages back to the source nodes, it inserts the 100 unestimated pilots of h_1 and h_2 into the frame. Therefore, among 300 subcarriers, 50 subcarriers are assigned to the pilots to estimate the relay to source channel in the second time slot, 100 subcarriers are assigned to the unestimated pilots from the relay to the source, and the rest 150 subcarriers are assigned for data transmission. Thus, in order to estimate the source-to-relay channel h1 and h2 in the first time slot and feed this information back to the source nodes in the second time slot, about 50% of the bandwidth is used to feed back the channel state information. The advantage of this method is that we achieve the synchronization of messages and channel states. The disadvantage is that we lose half of the bandwidth and utilize them to feed the

channel state information from the relay back to the source nodes. From this tradeoff we see that the efficiency of two-way relay system maybe severally degraded due to implementation issues.

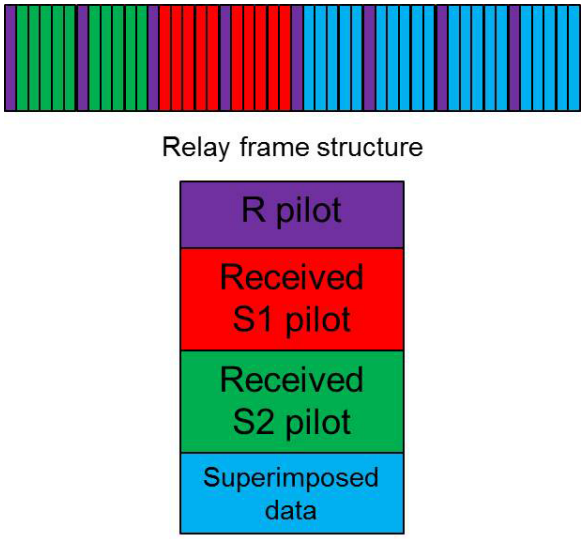

Fig. 19.: The Channel State Information at the Sources

F. Interference Cancellation at Source Nodes

In the two-way relay system, the self-interference cancellation assumes that the sources have perfect knowledge of both the source-to-relay and relay-to-source channels. As mentioned in the previous section, the channel states of the source to relay channels are fed back to the source by inserting the unestimated pilots into the broadcasted message. After each source receives the broadcasted messages from the relay node in the second time slot, it extracts the underestimated pilots h_1 and h_2, respectively. For example, after S1 receives the messages from the relay, it first extracts the pilots in the received signals to estimate the relay-to-source channel state in the second time slot. After cancelling this channel gain, it extracts the source-to-relay channel pilots which are inserted in the broadcasted messages. Finally, the S1 cancels the

self-interference and decodes the message from S1. The case for S2 is symmetric to the steps at S1.

G. Source to Source Synchronization

The source-node synchronization is done at the beginning of the two-way relay communication scheme, which requires the source synchronization error to be within 20 nanoseconds. In particular, the synchronization is achieved by using the PXI trigger within the FPGA to trigger the message flow from FPGA to baseband transceiver. In order to achieve this harsh requirement in our system, the two source nodes are configured within the same chassis, and the PXI trigger within the chassis is deployed to trigger the source nodes at the same time. The PXI trigger is controlled by the RT controller, and it is sent simultaneously to both source nodes. There are other methods such as using GPS to solve the source to source synchronization problem, which are usually much more costy (but necessary if we want to distribute the two source nodes at different locations). There are several steps for the two source nodes to be synchronized. The first step is to lock the two source nodes to the same clock. Then the data transmission time of both source nodes need to be aligned. However, there is a non-deterministic jitter and delay within the RT controller. Thus, a hardware trigger mechanism is proposed to trigger the signal transmission in the FPGA. As shown in Fig. 20, we utilize a FIFO queue to obtain the aligned messages. At the beginning, two data streams are fed into the signal processing blocks in the FPGAs and they are not aligned. Then the FPGAs are locked to the same clock, i.e., the operating speed of the FPGAs are the same. Once the signal processing is finished, each data stream is pushed into a FIFO queue in the FPGAs. We set a threshold for each queue, and then use the hardware triggers within the FPGA to release the data

to the output interface once the thresholds have been reached at both FPGAs. With such a method, we achieve the synchronization between two source nodes.

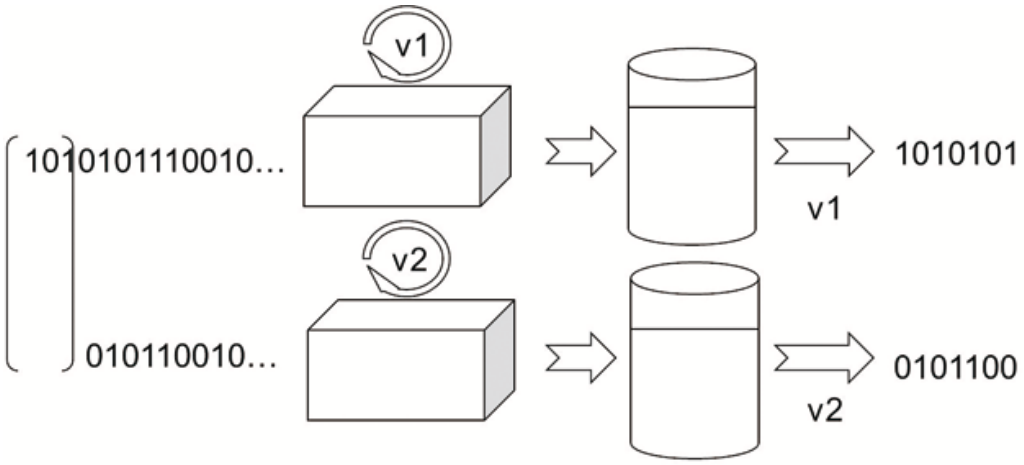

Fig. 20.: Principle of Source to Source Synchronization in FPGA

H. Source and Relay Synchronization

The source and relay chassis also need to be synchronized, since the two-way relay system needs the timing alignment between the TX and RX time slots to achieve reliable communications. The requirement of this synchronization is on the milliseconds level. As proposed in the previous chapter, we apply the master and slave mechanism to solve the source and relay synchronization, which requires an additional communication channel to coordinate both chassis. As shown in Fig. 21, the source and relay chassis are connected with each other by an Ethernet cable. As an example, once the TX time slot of source is ready to start, the source chassis sends a trigger signal to the relay through the Ethernet cable, and it triggers the relay chassis to start the reception and send a confirmation message back to the source chassis. After the source chassis receives the confirmed message, the TX time slot of source chassis starts.

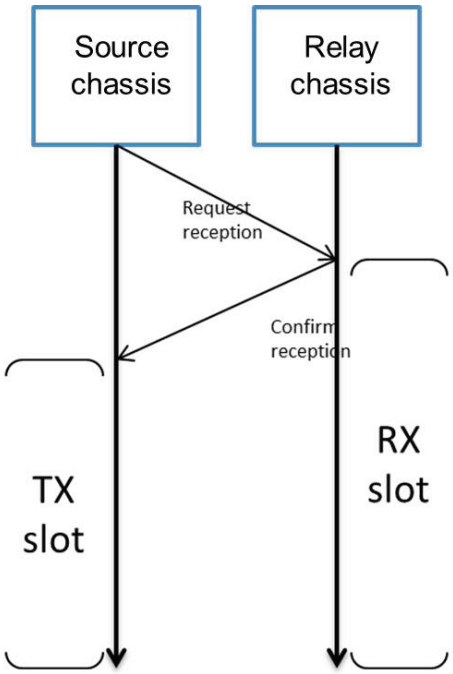

Fig. 21.: Source to Relay Handshaking Protocol

I. OFDM Frame Synchronization

The frame detection algorithm is to detect the beginning of the OFDM symbol. In our two-way relay system, a maximum likelihood estimation algorithm is implemented. The complexity is O(N^2) where N is the number of symbols within each OFDM frame. The detection algorithm has to be calculated in less than 1 millisecond in order to meet the real-time receiver requirement. In our systems, we implement this maximum likelihood estimation algorithm in the FPGA, and we optimize the complexity to O(N). The process of our ML estimation scheme is described below. The synchronization algorithm explores the structure of OFDM symbol with long CP. For example, as shown in Fig. 22, for a 5MHz OFDM symbol, the long CP length is 128 and the number of FFT points is 512. Therefore, given a sequence of OFDM frames, the correlation of the long CP is calculated to detect the beginning of the

OFDM symbol. The Beginning index (BI) is calculated according to the following formula [3]:

$$BI = argmax_{x\in[0,639]} \left\{ \sum_{n=0}^{127} N(x+n)N^*(x+n+512) \right\}. \tag{4.1}$$

Currently the OFDM frame synchronization algorithm is implemented in the FPGA and it finishes in several microseconds.

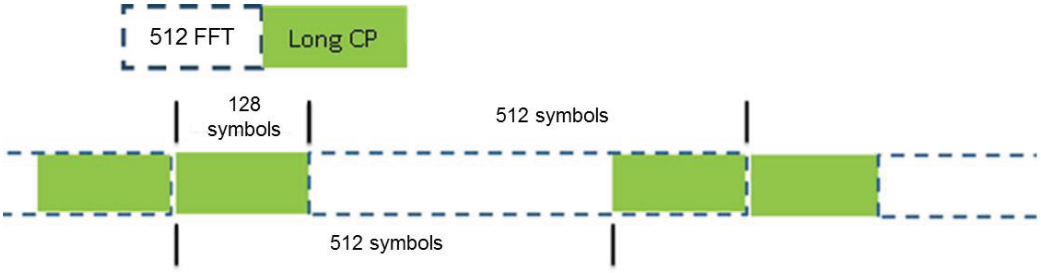

Fig. 22.: OFDM Frame Synchronization

CHAPTER V

CONCLUSION

In this thesis, I prototyped the two-way relay system based on analog network coding. The transmitter was implemented in the FPGA the receiver was implemented in the real-time controller. The physical layer mimics the 5MHz LTE downlink scenario. Several synchronization methods were provided to solve the frame synchronization of OFDM symbols, timing alignment of the two source nodes, and handshaking between the source and relay nodes. A new channel estimation scheme was proposed to estimate the source to relay channels at the same time. Finally, I demonstrated that the two-way relay system could achieve reliable communications with successful self-interference cancellation, but with significant bandwidth losses due to the system overhead for estimating channel state information and achieving synchronization.

REFERENCES

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow", IEEE Trans. Inf. Theory, vol. 46, no. 4, pp. 1204-1216, Jul. 2000.

[2] R. Zhang, Y.-C. Liang, C. C. Chai, and S. Cui, "Optimal beamforming for two-way multi-antenna relay channel with analogue network coding",IEEE J. Sel. Areas Commun., vol. 27, pp. 699-712, June 2009.

[3] J.-J. van de Beek, M. Sandell, and P. O. Borjesson, "ML estimation of time and frequency offset in OFDM systems", IEEE Trans. Signal Processing, vol. 45, pp. 1800-1805, July 1997.

[4] Jim Zyren, "Overview of the 3GPP Long Term Evolution Physical Layer", white paper, Freescale Semiconductor, Austin TX, 2007.

[5] National Instruments, Available: http://www.ni.com, June 2012.

VITA

Qiong Wu received a B.S. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 2010. He will be earned his Master degree in Texas A&M University, College Station, TX, USA, in 2012. He has been working on the hardware prototyping project of the two-way relay system during the study in Texas A&M University. Mr. Wu can be reached at 214 Zachry Engineering Center, TAMU 3128 College Station, Texas 77843-3128.

The typist for this thesis was Qiong Wu.